Available online at www.sciencedirect.com

SCIENCE DIRECT®

Theoretical Computer Science 354 (2006) 169–172

Theoretical Computer Science

www.elsevier.com/locate/tcs

TACAS 2003 Special Issue—Preface

Research in the field of computer-aided verification has made tremendous strides during the past decade. Throughout this period of progress, the International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS) has been one of the premier venues for reporting on results in this field. In this special issue of Theoretical Computer Science (TCS), we are pleased to present selected papers from the 9th TACAS held during April 2003 in Warsaw, Poland as part of the Joint European Conferences on Theory and Practice of Software (ETAPS).

In 2003, TACAS received a record number of submissions (160 papers) out of which 25.6% only were kept by the conference program committee after a rigorous selection process. After the conference Don Sannella (editor-in-chief of this journal) proposed that we put together a special issue of TCS dedicated to selected papers from TACAS 2003, while Bernhard Steffen (co-editor-in-chief of the International Journal for Software Tools for Technology Transfer (STTT)) similarly proposed that we prepare a special section of STTT. Although STTT had traditionally published a special issue for previous versions of TACAS, both editors and TACAS Program Chairs felt that TACAS 2003 had enough strong papers to fulfill both requests—with a clear separation drawn between the type of papers selected for TCS and those selected for STTT.

For STTT, we only retained those TACAS 2003 papers supported with a significant software implementation, the maturity of which should be sufficient for a rapid technology transfer. We invited 6 papers out of 41 and, after the reviewing process, only 5 papers were finally selected for the STTT issue. For this TCS issue, we invited 7 papers of 41 that dealt with more foundational or theoretical issues, and after the reviewing process, each of these invited papers was accepted for this special issue.

The diversity of techniques considered by papers selected for this issue—ranging across decision procedures for various logics, automata models for real-time and hybrid systems, foundations for task scheduling, use of game theory for verification, and various forms of compositional reasoning reflect the maturity and breadth of the field of computer-aided verification. We believe all these papers project the essence of TACAS—they present rigorous techniques for developing tools that can be applied with good effect to reason about practical problems. Many of these papers do have associated tool implementations and several papers report on experimental studies that demonstrate the practicality of approaches they propose.

Special issue contents

Foundations of temporal logic model checking: Temporal logics such as Linear Temporal Logic (LTL) and Computational Tree Logic (CTL) are often used as property specification languages for concurrent and reactive systems. The μ-calculus has often described as a "machine language" for temporal logics because it is an extremely general modal logic including both universal and existential path quantification into which many temporal logics such as LTL, CTL*, and dynamic logics such as Propositional Dynamic Logic (PDL) can be translated. One source of computational complexity when using the μ-calculus for automated reasoning is its ability to specify an unbounded number of switches between universal and existential branching modes.

0304-3975/$ - see front matter
doi:10.1016/j.tcs.2005.11.014

In their paper "On the Universal and Existential Fragments of the μ-Calculus", Henzinger, Kupferman, and Majumdar study classic decision problems for the μ-calculus in universal fragment of the logic (in which the existential branching mode is not allowed) and in the existential fragment (in which the universal branching mode is not allowed). One motivation for this study is that the universal fragment is rich enough to express most specifications of interest, and therefore improved algorithms for the universal fragment are of practical importance. The authors show that the classic problems of satisfiability and validity are indeed simpler when considering each fragment in isolation. However, this is not the case for the problems of model checking and implication. The authors also show the corresponding result for the alternation-free fragment of the μ-calculus. The contributions of this paper include a clean summary of relevant complexity results for different temporal logics. These results serve as an indication to other researchers regarding which problems they can hope to implement efficiently and which ones they cannot. Moreover, the results imply that efforts to find a polynomial-time model-checking algorithm for the μ-calculus can be replaced by efforts to find an algorithm for the universal or existential fragment.

Strategies for combining decision procedures: A decision procedure is an algorithm that can be used to determine whether a logical statement is a tautology. Decision procedures for fragments of first-order logic and arithmetic have long played an important role in theorem proving engines, and they are increasingly being used in different forms of automated program analysis and verification engines. It is often required to combine decision procedures for various theories to achieve a foundation for dealing with rich formalisms required for realistic applications. Implementing efficient algorithms for combining decision procedures has been a challenge, and the correctness of such combination methods is often precarious.

In their paper "Strategies for Combining Decision Procedures", Sylvain Conchon and Sava Krstić describe an inference system that has the classical Nelson–Oppen procedure at its core and includes several optimizations: variable abstraction with sharing, canonization of terms at the theory level, and Shostak's streamlined generation of new equalities for theories with solvers. The transitions of their system are fine-grained enough to model most of the mechanisms currently used in designing combination procedures. In particular, with a simple language of regular expressions they are able to describe several combination algorithms as strategies for their inference system, from the basic Nelson–Oppen to the very highly optimized one, recently given by Shankar and Ruess. The authors' approach for presenting the basic system at a high level of generality and non-determinism allows transparent correctness proofs that can be extended in a modular fashion when new features are introduced in the system.

Compositional analysis for parameterized systems: Many safety critical systems that have been considered by the verification community are parameterized by the number of concurrent components in the system, and hence describe an infinite family of systems. Traditional model checking techniques can only be used to verify specific instances of this family. In their paper "Compositional Analysis for Verification of Parameterized Systems", Basu and Ramakrishnan present a technique based on compositional model checking and program analysis that provides automatic verification of infinite families of systems. The technique views a parameterized system as an expression in Milner's process algebra CCS and interprets this expression over a domain of formulas (modal μ-calculus), considering a process as a property transformer. The transformers are constructed using partial model checking techniques. At its core, the technique solves the verification problem by finding the limit of a chain of formulas. The authors present a widening operation to find such a limit for properties expressible in a subset of modal μ-calculus. The authors demonstrate the utility of their technique on a number of parameterized systems.

Modular strategies for game-theoretic verification: Many problems in formal verification and program analysis can be formalized as computing winning strategies for two-player games on graphs. In their paper, "Modular Strategies for Recursive Game Graphs", Alur, La Torre, and Madhusudan focus on solving games in recursive game graphs that can model the control in sequential programs with recursive procedure calls. While such games can be viewed as the pushdown games studied in the literature, the authors investigate the benefits of a restricted setting in which the natural notion of winning requires the strategies to be modular with only local memory; that is, resolution of choices within a module does not depend on the context in which the module is invoked, but only on the history within the current invocation of the module. While reachability in (global) pushdown games is known to be Exptime-complete, the authors show reachability in modular games to be NP-complete. A fixed-point computation algorithm for solving

modular games is defined such that in the worst-case the number of iterations is exponential in the total number of returned values from the modules. If the strategy within a module does not depend on the global history, but can remember the history of the past invocations of this module, that is, if memory is local but persistent, the authors show that reachability becomes undecidable.

Predicate abstraction for hybrid systems: To make model checking tractable, it is often necessary to create more abstract (less detailed) versions of system models to reduce the size of the state-space being analyzed. Predicate abstraction has emerged to be a powerful automatic abstraction technique for extracting finite-state models from infinite-state systems. Recently, predicate abstraction has been shown to enhance the effectiveness of reachability computation techniques for *hybrid systems*—systems with both discrete and continuous variables that are often used to model computer control of physical devices. Given a hybrid system with linear dynamics and a set of linear predicates, a verifier using predicate abstraction performs an on-the-fly search of the finite discrete quotient whose states correspond to the truth assignments to the input predicates. The success of this approach depends on the choice of the predicates used for abstraction.

In their paper "Counter-Example Guided Predicate Abstraction of Hybrid Systems", Alur, Dang, and Ivančić show how to adapt well-known concepts used in predicate abstraction for discrete systems to hybrid systems. Specifically, they focus on identifying predicates automatically by analyzing spurious counter-examples generated by the search in the abstract state-space. They present the basic techniques for discovering new predicates that will rule out closely related spurious counter-examples, optimizations of these techniques, implementation of these in a verification tool, and case studies demonstrating the promise of the approach.

Using timed automata for scheduling: Model checking is usually thought of as a verification technique. However, recent work is emphasizing the use of model checking tools to explore a space of schedules for multiple tasks to be performed—with the goal of identifying schedules that satisfy desirable properties (e.g., earlier overall completion time). In classic scheduling theory, real-time tasks are usually assumed to be periodic, i.e., tasks are released and computed with fixed rates periodically. To relax the stringent constraints on task arrival times, researchers have proposed using timed automata to describe task arrival patterns.

Abdeddaïm, Asarin, and Maler begin their paper "Scheduling with Timed Automata" with a nice introduction that outlines the general problem of scheduling and motivates the use of timed automata as a general-purpose and more flexible alternative to traditional approaches taken in the area of Operations Research. They lay the groundwork for future explorations by showing how efficient shortest path algorithms for timed automata can find optimal schedules for the classical job-shop scheduling problem. The authors then introduce a concept of non-lazy schedules that allows one to restrict attention to a finite subset of the non-countable set of possible schedules. The second part of the paper considers an extension of the job-shop problem in which task durations admit a bounded uncertainty. After defining the appropriate criterion of optimality, an algorithm is developed in the dynamic programming style, which finds adaptive scheduling strategies that are optimal in this sense.

Also considering the application of timed automata to scheduling of real-time systems, Fersman, Mokrushin, Pettersson, and Yi address a key issue in this approach to scheduling: the number of timed-automata clocks needed in the analysis is proportional to the maximal number of schedulable task instances associated with a model, which in many cases is huge. In their paper "Schedulability Analysis of Fixed-Priority Systems Using Time Automata", they show that for fixed priority scheduling strategy, the schedulability checking problem can be solved using standard timed automata with *two* extra clocks in addition to the clocks used in the original model to describe task arrival times. The analysis can be done in a similar manner to response time analysis in classic Rate-Monotonic Analysis (RMA). The result is further extended to systems with data-dependent control, in which the release time of a task may depend on the time-point at which other tasks finish their execution. For the case when the execution times of tasks are constants, the authors show that the schedulability problem can be solved using $n + 1$ extra clocks where n is the number of tasks. The presented analysis techniques have been implemented in the TIMES tool. For systems with only periodic tasks, the performance of the tool is comparable with tools implementing the classic RMA technique based on equation-solving, without suffering from the exponential explosion in the number of tasks.

Conclusion

We would like to thank heartily the referees for their expertise and their commitment to the quality of the present special issue. At this point, we hope that these brief accounts have convinced you to plunge into this special issue of TCS and we wish you a fruitful reading.

Hubert Garavel
INRIA Rhône-Alpes / VASY,
655, avenue de l'Europe,
38330 Montbonnot Saint-Martin, France
E-mail addresses: hubert.garavel@inria.fr

John Hatcliff
Kansas State University, 234 Nichols Hall,
Manhattan, KS 66506, USA
E-mail addresses: hatcliff@cis.ksu.edu
TACAS 2003 Co-Chairs